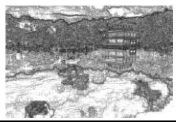



Image gradients and edges

Thurs Sept 3

Prof. Kristen Grauman
UT-Austin

Last time

- Various models for image "noise"
- Linear filters and convolution useful for
  - Image smoothing, removing noise
    - Box filter
    - Gaussian filter
    - Impact of scale / width of smoothing filter
- Separable filters more efficient
- Median filter: a non-linear filter, edge-preserving

Image filtering

- Compute a function of the local neighborhood at each pixel in the image
  - Function specified by a "filter" or mask saying how to combine values from neighbors.
- Uses of filtering:
  - Enhance an image (denoise, resize, etc)
  - Extract information (texture, edges, etc)
  - Detect patterns (template matching)

Today

Adapted from Derek Hoiem

Edge detection

- **Goal**: map image from 2d array of pixels to a set of curves or line segments or contours.
- **Why?**

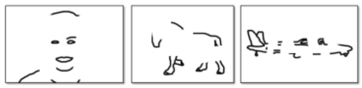
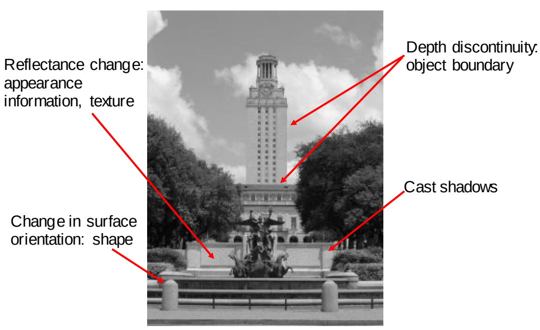


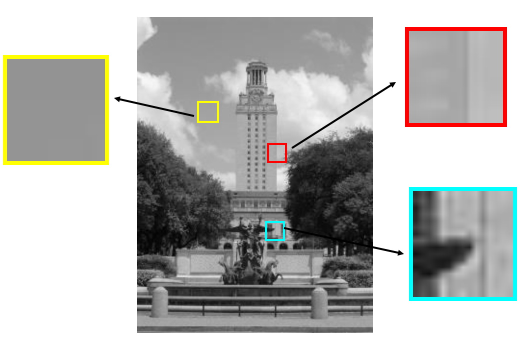
Figure from J. Shotton et al, PAMI 2007

- **Main idea**: look for strong gradients, post-process

What causes an edge?

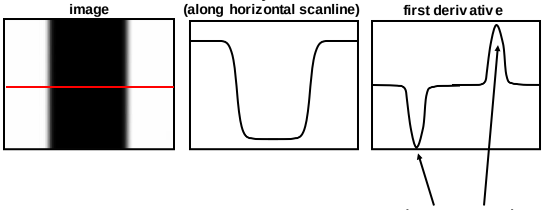


Edges/gradients and invariance



Derivatives and edges

An edge is a place of rapid change in the image intensity function.



Source: L. Lazebnik

Derivatives with convolution

For 2D function, f(x,y), the partial derivative is:

$$\frac{\partial f(x,y)}{\partial x} = \lim_{\varepsilon \to 0} \frac{f(x+\varepsilon, y) - f(x,y)}{\varepsilon}$$

For discrete data, we can approximate using finite differences:

$$\frac{\partial f(x,y)}{\partial x} \approx \frac{f(x+1,y) - f(x,y)}{1}$$

To implement above as convolution, what would be the associated filter?

Partial derivatives of an image

$$\frac{\partial f(x,y)}{\partial x}$$

$$\frac{\partial f(x,y)}{\partial y}$$

-1	1

?

-1
1

or

1
-1

Which shows changes with respect to x?

(showing filters for correlation)

Assorted finite difference filters

Prewitt: $M_x = \begin{array}{|c|c|c|}\hline -1 & 0 & 1 \\ \hline -1 & 0 & 1 \\ \hline -1 & 0 & 1 \\ \hline \end{array}$; $M_y = \begin{array}{|c|c|c|}\hline 1 & 1 & 1 \\ \hline 0 & 0 & 0 \\ \hline -1 & -1 & -1 \\ \hline \end{array}$

Sobel: $M_x = \begin{array}{|c|c|c|}\hline -1 & 0 & 1 \\ \hline -2 & 0 & 2 \\ \hline -1 & 0 & 1 \\ \hline \end{array}$; $\boxed{M_y = \begin{array}{|c|c|c|}\hline 1 & 2 & 1 \\ \hline 0 & 0 & 0 \\ \hline -1 & -2 & -1 \\ \hline \end{array}}$

Roberts: $M_x = \begin{array}{|c|c|}\hline 0 & 1 \\ \hline -1 & 0 \\ \hline \end{array}$; $M_y = \begin{array}{|c|c|}\hline 1 & 0 \\ \hline 0 & -1 \\ \hline \end{array}$

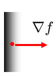
```
>> My = fspecial('sobel');
>> outim = imfilter(double(im), My);
>> imagesc(outim);
>> colormap gray;
```

Image gradient

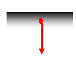
The gradient of an image:

$$\nabla f = \left[\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right]$$

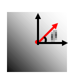
The gradient points in the direction of most rapid change in intensity



$\nabla f = \left[\frac{\partial f}{\partial x}, 0\right]$



$\nabla f = \left[0, \frac{\partial f}{\partial y}\right]$



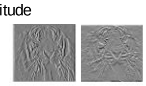
$\nabla f = \left[\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right]$

The **gradient direction** (orientation of edge normal) is given by:

$$\theta = \tan^{-1}\left(\frac{\partial f}{\partial y} / \frac{\partial f}{\partial x}\right)$$

The **edge strength** is given by the gradient magnitude

$$\|\nabla f\| = \sqrt{\left(\frac{\partial f}{\partial x}\right)^2 + \left(\frac{\partial f}{\partial y}\right)^2}$$



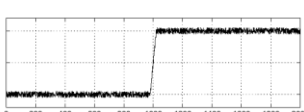
Slide credit Steve Seitz

Effects of noise

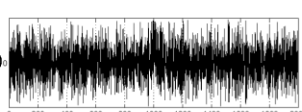
Consider a single row or column of the image

- Plotting intensity as a function of position gives a signal

$f(x)$



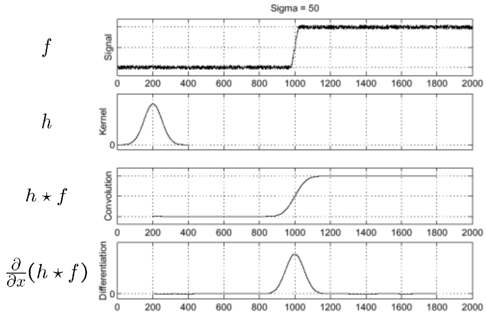
$\frac{d}{dx}f(x)$



Where is the edge?

Slide credit Steve Seitz

Solution: smooth first

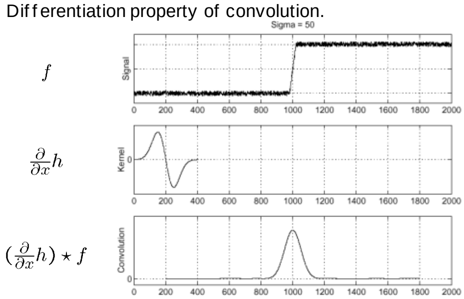


Where is the edge?

Derivative theorem of convolution

$$\frac{\partial}{\partial x}(h \star f) = \left(\frac{\partial}{\partial x}h\right) \star f$$

Differentiation property of convolution.

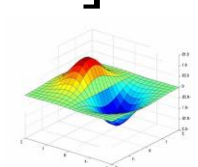


Slide credit Steve Seitz

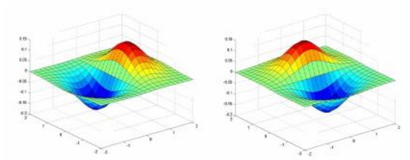
Derivative of Gaussian filters

$$(I \otimes g) \otimes h = I \otimes (g \otimes h)$$

$$\begin{bmatrix} 0.0030 & 0.0133 & 0.0219 & 0.0133 & 0.0030 \\ 0.0133 & 0.0596 & 0.0983 & 0.0596 & 0.0133 \\ 0.0219 & 0.0983 & 0.1621 & 0.0983 & 0.0219 \\ 0.0133 & 0.0596 & 0.0983 & 0.0596 & 0.0133 \\ 0.0030 & 0.0133 & 0.0219 & 0.0133 & 0.0030 \end{bmatrix} \otimes \begin{bmatrix} 1 & -1 \end{bmatrix}$$



Derivative of Gaussian filters



x-direction   *y*-direction

Source: L. Lazebnik

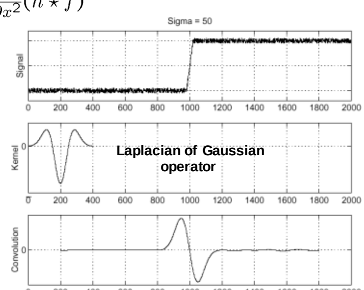
Laplacian of Gaussian

Consider $\frac{\partial^2}{\partial x^2}(h \star f)$

f

$\frac{\partial^2}{\partial x^2}h$

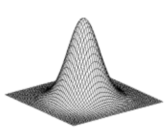
$(\frac{\partial^2}{\partial x^2}h) \star f$



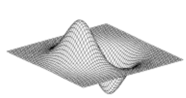
Where is the edge?

Slide credit: Steve Seitz

2D edge detection filters

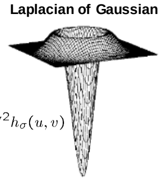


Gaussian
$h_\sigma(u,v) = \frac{1}{2\pi\sigma^2} e^{-\frac{u^2+v^2}{2\sigma^2}}$



derivative of Gaussian
$\frac{\partial}{\partial x} h_\sigma(u,v)$

Laplacian of Gaussian



$\nabla^2 h_\sigma(u,v)$

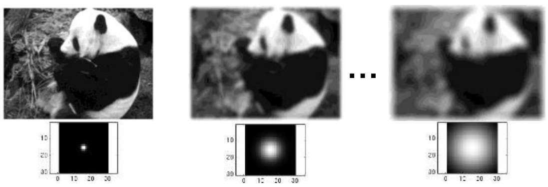
- ∇^2 is the Laplacian operator:

$$\nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}$$

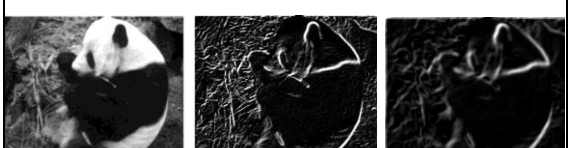
Slide credit: Steve Seitz

Smoothing with a Gaussian

Recall: parameter σ is the “scale” / “width” / “spread” of the Gaussian kernel, and controls the amount of smoothing.



Effect of σ on derivatives



σ = 1 pixel σ = 3 pixels

The apparent structures differ depending on Gaussian’s scale parameter.

Larger values: larger scale edges detected
Smaller values: finer features detected

So, what scale to choose?

It depends what we’re looking for.

Mask properties

- Smoothing
  - Values positive
  - Sum to 1 → constant regions same as input
  - Amount of smoothing proportional to mask size
  - Remove "high-frequency" components; "low-pass" filter
- Derivatives
  - __________ signs used to get high response in regions of high contrast
  - Sum to ___ → no response in constant regions
  - High absolute value at points of high contrast

Seam carving: main idea

Content-aware resizing

Intuition:

- Preserve the most "interesting" content
  → Prefer to remove pixels with low gradient energy
- To reduce or increase size in one dimension, remove irregularly shaped "seams"
  → Optimal solution via dynamic programming.

Seam carving: main idea

$$Energy(f) = \sqrt{\left(\frac{\partial f}{\partial x}\right)^2 + \left(\frac{\partial f}{\partial y}\right)^2}$$

- Want to remove seams where they won't be very noticeable:
  - Measure "energy" as gradient magnitude
- Choose seam based on **minimum total energy path** across image, subject to 8-connectedness.

Seam carving: algorithm

$$Energy(f) = \sqrt{\left(\frac{\partial f}{\partial x}\right)^2 + \left(\frac{\partial f}{\partial y}\right)^2}$$

Let a vertical seam **s** consist of *h* positions that form an 8-connected path.

Let the cost of a seam be: $Cost(\mathbf{s}) = \sum_{i=1}^{h} Energy(f(s_i))$

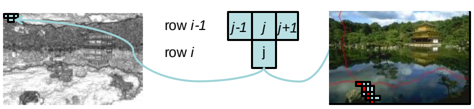
Optimal seam minimizes this cost: $\mathbf{s}^* = \min_{\mathbf{s}} Cost(\mathbf{s})$

Compute it efficiently with dynamic programming.

Seam carving: algorithm

- Compute the cumulative minimum energy for all possible connected seams at each entry *(i,j)*:

$$\mathbf{M}(i,j) = Energy(i,j) + \min\left(\mathbf{M}(i-1,j-1), \mathbf{M}(i-1,j), \mathbf{M}(i-1,j+1)\right)$$



Energy matrix (gradient magnitude)

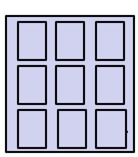
M matrix: cumulative min energy (for vertical seams)

- Then, min value in last row of **M** indicates end of the minimal connected vertical seam.
- Backtrack up from there, selecting min of 3 above in **M**.

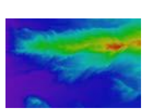
Example

$$\mathbf{M}(i,j) = Energy(i,j) + \min\left(\mathbf{M}(i-1,j-1), \mathbf{M}(i-1,j), \mathbf{M}(i-1,j+1)\right)$$

1	3	0
2	8	9
5	2	6



Energy matrix (gradient magnitude)

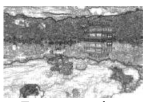


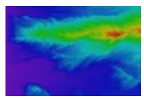
M matrix (for vertical seams)

Example

$$\mathbf{M}(i,j) = Energy(i,j) + \min\left(\mathbf{M}(i-1,j-1), \mathbf{M}(i-1,j), \mathbf{M}(i-1,j+1)\right)$$

$$\begin{matrix} \boxed{1} & 3 & 0 \\ \boxed{2} & 8 & 9 \\ 5 & \boxed{2} & 6 \end{matrix} \qquad \begin{matrix} \boxed{1} & 3 & 0 \\ \boxed{3} & 8 & 9 \\ 8 & \boxed{5} & 14 \end{matrix}$$


Energy matrix
(gradient magnitude)


M matrix
(for vertical seams)

Other notes on seam carving

- Analogous procedure for horizontal seams
- Can also insert seams to *increase* size of image in either dimension
  - Duplicate optimal seam, averaged with neighbors
- Other energy functions may be plugged in
  - E.g., color-based, interactive,…
- Can use combination of vertical and horizontal seams

Gradients -> edges

Primary edge detection steps:

1. Smoothing: suppress noise
2. Edge enhancement: filter for contrast
3. Edge localization

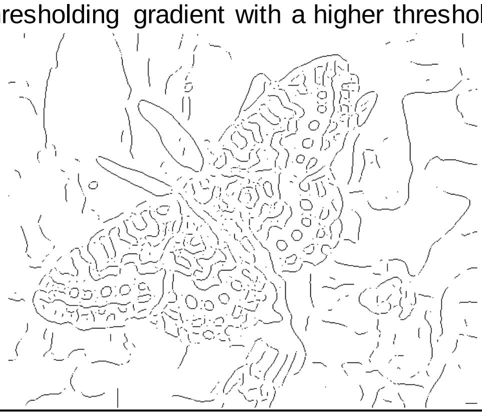
   Determine which local maxima from filter output are actually edges vs. noise

   - Threshold, Thin

Thresholding

- Choose a threshold value t
- Set any pixels less than t to zero (off)
- Set any pixels greater than or equal to t to one (on)

Thresholding gradient with a higher threshold



Canny edge detector

- Filter image with derivative of Gaussian
- Find magnitude and orientation of gradient
- **Non-maximum suppression**:
  - Thin wide "ridges" down to single pixel width
- **Linking and thresholding** (**hysteresis**):
  - Define two thresholds: low and high
  - Use the high threshold to start edge curves and the low threshold to continue them
- MATLAB: `edge(image, 'canny');`
- `>>help edge`

Source: D. Lowe, L. Fei-Fei

The Canny edge detector

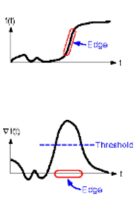
original image (Lena)

Slide credit: Steve Seitz

The Canny edge detector

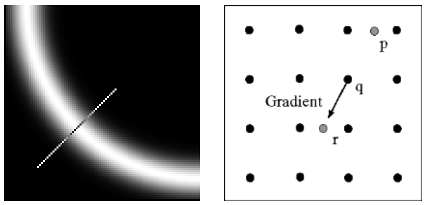
thresholding

The Canny edge detector



How to turn these thick regions of the gradient into curves?

Non-maximum suppression



Check if pixel is local maximum along gradient direction, select single max across width of the edge

- requires checking interpolated pixels p and r

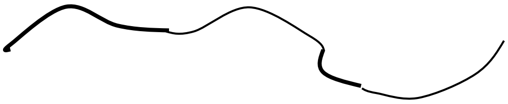
The Canny edge detector

Problem: pixels along this edge didn't survive the thresholding

thinning
(non-maximum suppression)

Hysteresis thresholding

- Use a high threshold to start edge curves, and a low threshold to continue them.



Source: Steve Seitz

Hysteresis thresholding

original image

high threshold
(strong edges)

low threshold
(weak edges)

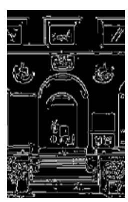
hysteresis threshold

Source: L. Fei-Fei

Hysteresis thresholding

high threshold
(strong edges)

low threshold
(weak edges)



hysteresis threshold

Source: L. Fei-Fei

Recap: Canny edge detector

- Filter image with derivative of Gaussian
- Find magnitude and orientation of gradient
- **Non-maximum suppression**:
  - Thin wide "ridges" down to single pixel width
- **Linking and thresholding** (**hysteresis**):
  - Define two thresholds: low and high
  - Use the high threshold to start edge curves and the low threshold to continue them
- MATLAB: `edge(image, 'canny');`
- `>>help edge`

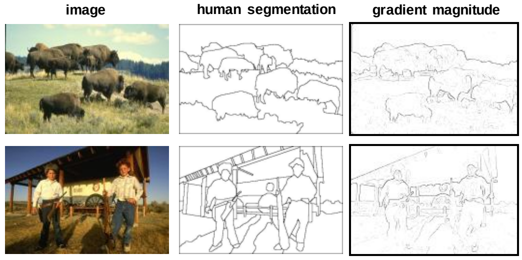
Source: D. Lowe, L. Fei-Fei

Low-level edges vs. perceived contours

Background   Texture   Shadows

Low-level edges vs. perceived contours

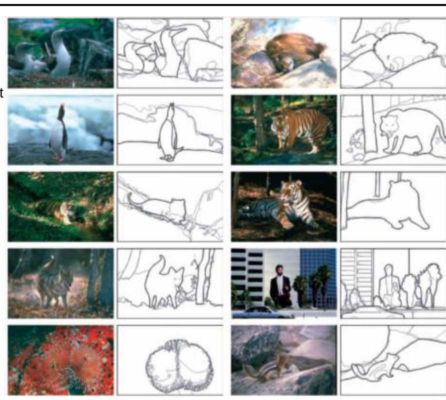
image   human segmentation   gradient magnitude



Berkeley segmentation database:
http://www.eecs.berkeley.edu/Research/Projects/CS/vision/grouping/segbench/

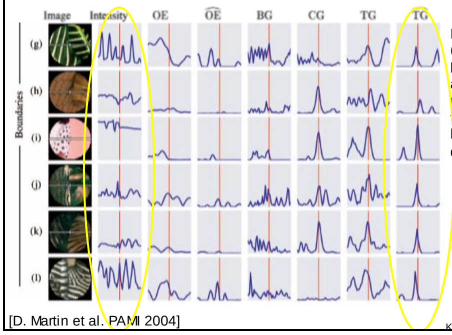
Source: L. Lazebnik

Learn from humans which combination of features is most indicative of a "good" contour?



[D. Martin et al. PAMI 2004]   Human-marked segment boundaries

What features are responsible for perceived edges?

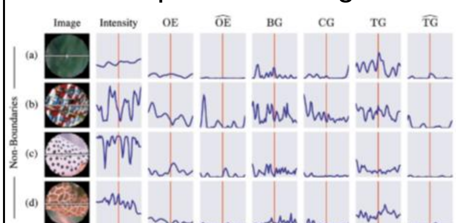


Feature profiles (oriented energy, brightness, color, and texture gradients) along the patch's horizontal diameter

[D. Martin et al. PAMI 2004]

Kristen Grauman UT-Austin

What features are responsible for perceived edges?



Feature profiles (oriented energy, brightness, color, and texture gradients) along the patch's horizontal diameter

[D. Martin et al. PAMI 2004]

Kristen Grauman UT-Austin

[D. Martin et al. PAMI 2004]

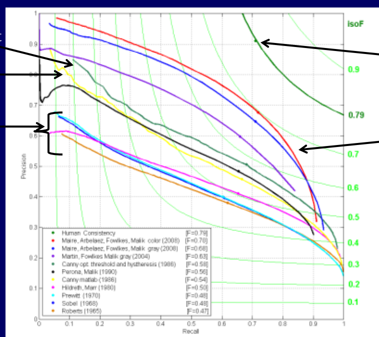
Kristen Grauman UT-Austin

Contour Detection

Canny+opt thresholds

Canny

Prewitt, Sobel, Roberts



Human agreement

Learned with combined features

UC Berkeley

Source: Jitendra Malik: http://www.cs.berkeley.edu/~malik/malik-talks-ptrs.html

Computer Vision Group

Recall: image filtering

- Compute a function of the local neighborhood at each pixel in the image
  - Function specified by a "filter" or mask saying how to combine values from neighbors.
- Uses of filtering:
  - Enhance an image (denoise, resize, etc)
  - Extract information (texture, edges, etc)
  - Detect patterns (template matching)

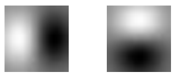
Adapted from Derek Hoiem

Filters for features

- Map raw pixels to an intermediate representation that will be used for subsequent processing
- Goal: reduce amount of data, discard redundancy, preserve what's useful

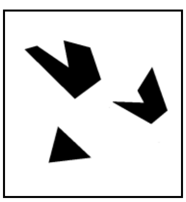
Template matching

- Filters as **templates**:
  Note that filters look like the effects they are intended to find --- “matched filters”



- Use normalized cross-correlation score to find a given pattern (template) in the image.
- Normalization needed to control for relative brightnesses.

Template matching

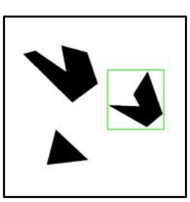


Scene

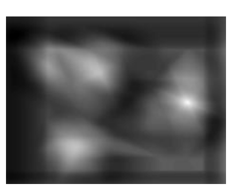
Template (mask)

A toy example

Template matching



Detected template



Correlation map

Recap: Mask properties

- Smoothing
  - Values positive
  - Sum to 1 → constant regions same as input
  - Amount of smoothing proportional to mask size
  - Remove "high-frequency" components; "low-pass" filter
- Derivatives
  - Opposite signs used to get high response in regions of high contrast
  - Sum to 0 → no response in constant regions
  - High absolute value at points of high contrast
- Filters act as templates
  - Highest response for regions that "look the most like the filter"
  - Dot product as correlation

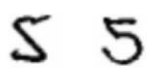


Fig. 1. Examples of two handwritten digits. In terms of pixel-to-pixel comparisons, these two images are quite different, but to the human observer, the shapes appear to be similar.

Figure from Belongie et al.

Chamfer distance

- Average distance to nearest feature

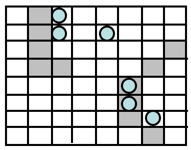
$$D_{chamfer}(T, I) \equiv \frac{1}{|T|} \sum_{t \in T} d_I(t)$$

$I =$ Set of points in image

$T =$ Set of points on (shifted) template

$d_I(t) =$ Minimum distance between point t and some point in I

Chamfer distance



$$D_{chamfer}(T, I) \equiv \frac{1}{|T|} \sum_{t \in T} d_I(t)$$

Chamfer distance

- Average distance to nearest feature

$$D_{chamfer}(T, I) \equiv \frac{1}{|T|} \sum_{t \in T} d_I(t)$$

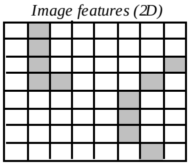
How is the measure different than just filtering with a mask having the shape points?

How expensive is a naïve implementation?

Edge image

Distance transform

Image features (2D)



Distance Transform

1	0	1	2	3	4	3	2
1	0	1	2	3	3	2	1
1	0	1	2	3	2	1	0
1	0	0	1	2	1	0	1
2	1	1	2	1	0	1	2
3	2	2	2	1	0	1	2
4	3	3	2	1	0	1	2
5	4	4	3	2	1	0	1

Distance Transform is a function $D(\cdot)$ that for each image pixel p assigns a non-negative number $D(p)$ corresponding to distance from p to the nearest feature in the image I

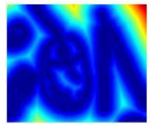
Features could be edge points, foreground points,…

Source: Yuri Boykov

Distance transform

original

edges



distance transform

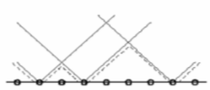
Value at (x,y) tells how far that position is from the nearest edge point (or other binary mage structure)

```
>> help bwdist
```

Distance transform (1D)

Two pass O(n) algorithm for 1D L_1 norm

1. Initialize: For all j
   D[j] ← 1_P[j] // 0 if j is in **P**, infinity otherwise
2. Forward: For j from 1 up to n-1
   D[j] ← min(D[j],D[j-1]+1)
3. Backward: For j from n-2 down to 0
   D[j] ← min(D[j],D[j+1]+1)

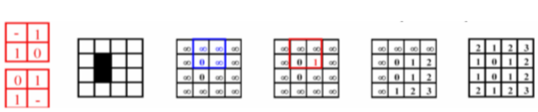


∞	0	∞	0	∞	∞	∞	0	∞
∞	0	1	0	1	2	3	0	1
1	0	1	0	1	2	1	0	1

Adapted from D. Huttenlocher

Distance Transform (2D)

- 2D case analogous to 1D
  - Initialization
  - Forward and backward pass
    - Fwd pass finds closest above and to left
    - Bwd pass finds closest below and to right

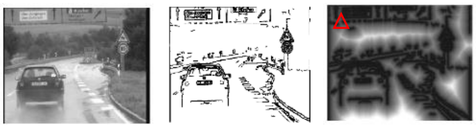


Adapted from D. Huttenlocher

Chamfer distance

- Average distance to nearest feature

$$D_{chamfer}(T, I) \equiv \frac{1}{|T|} \sum_{t \in T} d_I(t)$$



Edge image Distance transform image

Chamfer distance

Edge image Distance transform image

Fig from D. Gavrila, DAGM 1999

Chamfer distance: properties

- Sensitive to scale and rotation
- Tolerant of small shape changes, clutter
- Need large number of template shapes
- Inexpensive way to match shapes

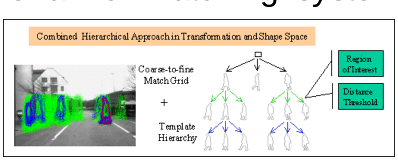
Chamfer matching system

- Gavrila et al. http://gavrila.net/Research/Chamfer_System/chamfer_system.html

Chamfer matching system

- Gavrila et al. http://gavrila.net/Research/Chamfer_System/chamfer_system.html

Chamfer matching system



- Gavrila et al. http://gavrila.net/Research/Chamfer_System/chamfer_system.html

Summary

- Image gradients
- Seam carving – gradients as “energy”
- Gradients → edges and contours
- Template matching
  - – Image patch as a filter
  - – Chamfer matching
    - Distance transform

Coming up

- A1 out, due in 2 weeks

- Tues: Binary image analysis
  - – Guest Lecture : Dr. Danna Gurari
- Thurs: Images/videos and text
  - – Guest Lecture: Prof. Ray Mooney